Étienne Dupont
Antoine Pallois

Institut Supérieur de l'Aéronautique et de l'Espace
SUPAERO

October 7, 2018

F-SA302 - SYSTÈMES AUTONOMES : PERCEPTION ET NAVIGATION

Simultaneous Localisation and Mapping (SLAM)

1 Introduction

During the lab session, we filled in a pre-written algorithm (see attached files) to simulate a 2D-SLAM robot using an extended Kalman filter. The algorithm's effectiveness lies in the parameters that influence the modelling of the SLAM process. We are now going to discuss the impact of those different parameters:

1. If we increase the control noise, what happens? And why?
2. If we increase the measurement noise, what happens? And why?
3. If we change the initial robot position, what happens? And why?
4. If we increase the initial robot position uncertainty, what happens? And why?
5. Why do we need to make sure that the innovation angle z(2) is centred around zero in section c6?
6. If we make the variable 'mapsize' smaller, what happens? And why?
7. Could you explain which parts of this code you should improve or modify if you were to use this code in a real robot, with real data?

2 Influence of the algorithm's parameters

2.1 Control noise influence

Experimentally and after playing for a while with the 2D-SLAM algorithm, we found an optimal value of $\sim 2 \cdot 10^{-2}$ for both parameters of the system noise standard deviation. If we lower these values, we tend to obtain divergence in the model: indeed, if we reduce the system noise too much, then the actual real states do not lie in the model's confidence boundaries. This means that the variance of the control noise is underestimated, leading to an exaggerated confidence in the control parameter, when estimating states.

On the contrary, if we increase the control noise, we obtain a lowered precision of the robot and the landmarks states estimations. Indeed, each "movement step" increases the uncertainties according to the control noise. As the extended Kalman filter will then put more and more trust in the measurements to do the estimation, this often brings a less reliable result. **It is thus essential to find to the balanced control noise, in order not to fall into either extreme.**

2.2 Measurement noise influence

We found out that a reasonable choice for the measurement noise's standard deviation would be 10 cm for the distance measurement noise and 1° for the angle measurement noise. The influence of these parameters is quite similar to the influence of the control noise, though we found that the landmarks' states uncertainty more strongly depends on the measurement noise than on the control noise. Reducing these value will result in a divergence with real states leaving the model's confidence boundary. On the contrary, increasing those parameters will lead to less reliable results with big uncertainties on both robot and landmarks states.

2.3 Robot's initial position influence

In the simulation, the "real" initial robot state is $[0; 0; 0]$, although in reality we cannot know the exact position of the robot. Changing the initial position to another one will simply offset both the robot's and the landmarks' positions during the following time steps: this initial error cannot be lowered by further measurements and control signals because it is due to an arbitrary choice of an absolute initial condition. However, when practising real SLAM, this would not be a problem because the interesting parameter is the relative position between the robot and the landmarks, which is preserved even when changing the initial position.

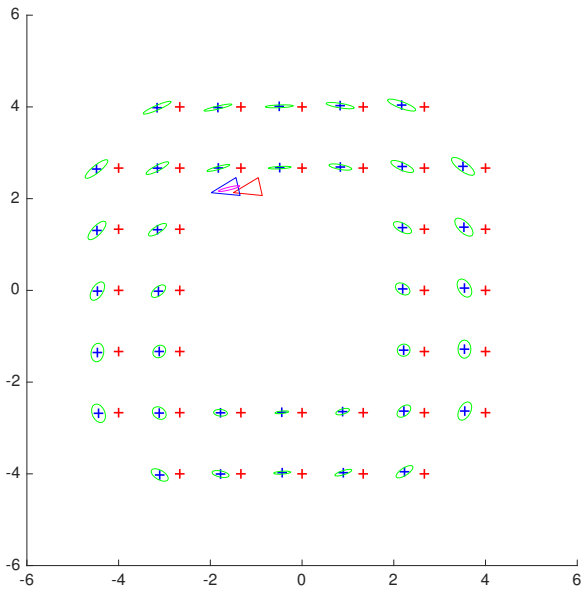

Figure 1: Simulation with an additional offset to the initial robot position: $[x = -0.5; y = 0; \alpha = 0]$. One clearly identifies the x-axis offset resulting from this initial error.

2.4 Robot's initial position uncertainty influence

In usual circumstances, we consider that at the start we know exactly the initial position of the robot; we thus consider that the initial covariance is null. However, if we change this and add for example an incertitude on the initial position of the robot $\sigma_x = \sigma_y = \sqrt{0.1}$ m and $\sigma_\alpha = \sqrt{0.1}$ rad (the standard deviation is then linked to the covariance matrix $P(r, r) = 0.1 \cdot \text{eye}(3, 3)$), we get an offset on the robot's position incertitude, as shown on Figure 2. In our case, adding the initial covariance corresponds to adding an offset on the robot's position incertitude, as we are adding an incertitude on an arbitrary choice of the absolute initial condition. Again, this would not be a problem when practising real SLAM, as the significant parameter would then be the relative

position between the robot and the landmarks, which is preserved when changing the initial incertitude. In real SLAM, the incertitude on the position of the robot would diminish over time, and not stay constant.

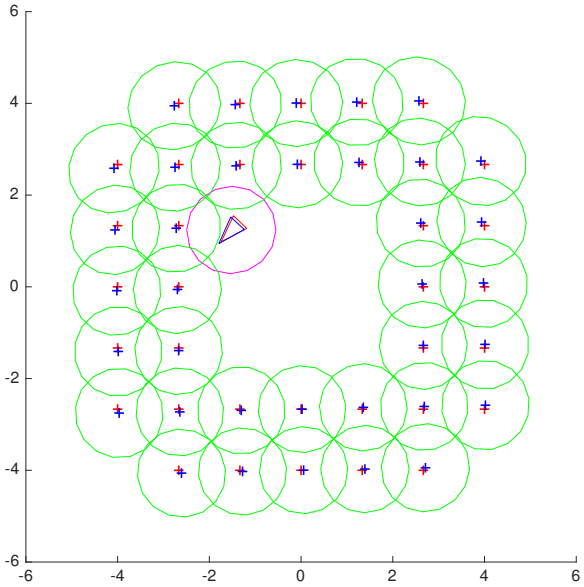

Figure 2: Simulation with an initial robot position incertitude of $P(r,r) = 0.1 \cdot \text{eye}(3,3)$. One clearly identifies the incertitude offset resulting from this initial covariance term.

2.5 Necessity for innovation angle centring

We need to have an innovation angle centred around zero and not multiples of 2π. Indeed, the Kalman Filter provides a correction which is proportional to the innovation. If we have multiples of 2π that are not zero for the innovation angle - which would still mean that estimations and measurements agree and there is no need to correct the estimation - the Kalman Filter will correct the angle proportionally to this innovation angle : we don't want this to happen.

2.6 Mapsize influence

The variable "Mapsize" corresponds to the space allocated to store the state of the robot and the landmarks. In our model, we are given a constant number of landmarks, which is 36. As the state of the robot occupy 3 dimensions and the state of one landmark occupy 2 dimensions, storing n landmarks will necessitate a mapsize $3 + 2n$, ranging in $[3, 75]$. Exceeding 75 for the mapsize would only result in more processing time for the algorithm. Decreasing this value from 75 to 3, we notice a gradual increase in the uncertainties for both robot and landmarks states. Indeed, the precision in the robot's state increases with the number of landmarks, because each observation of a known landmark reduces the uncertainty on the robot's state. Conversely, the uncertainty on a landmark's state depends on the uncertainty on the robot's state : the more precise the robot's state, the lower the uncertainty on the landmark's state.

2.7 Implementation in a real robot, with real data

For the implementation in a real robot and with real data, it is first of all essential to input the real system noise and real measurement noise. Indeed, these are known parameters as they are calculated by the constructors

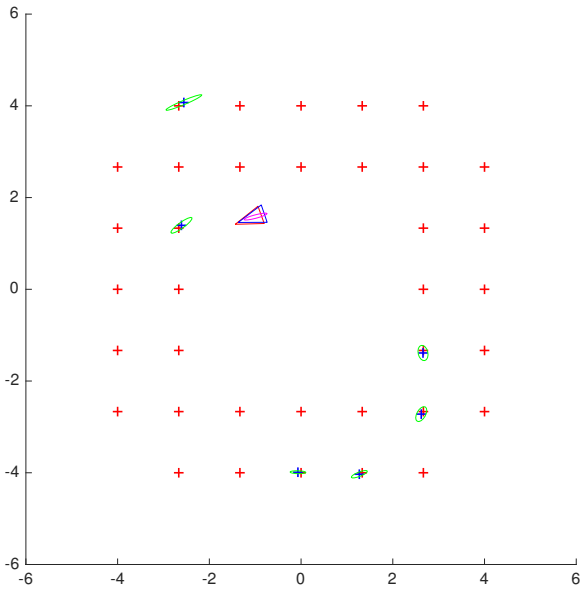

Figure 3: Simulation with a mapsize of 15, where only 6 landmarks are taken into account. One clearly sees that not all landmarks are used, as the system does not have place to store it.

and will give a good order of magnitude of what will need to be input.

Second of all, a hybrid positioning model using other types of sensors would be very useful. For instance, using the information given by a GPS sensor or a gyroscope will give more precise information on the location of the robot, thus helping to reduce even more the incertitude on its position. In general, having more than one solution to help the positioning of the robot will help to minimise the position incertitude.

Furthermore, the choice of the landmarks is essential in the effectiveness of the SLAM positioning method; having landmarks that can be easily confused will directly lead to increased and accumulated inaccuracies. **It is essential that the landmarks be significantly different and that they be regularly spaced in the area we need to cover.**

Programming a 3D version of the SLAM method will also be important for drones and their 3D evolution in the air. This will have numerous applications such as the localisation and the mapping of and inside buildings. This leads to another essential issue, which corresponds to the mapsize. Having too big a mapsize will generate tremendous computation time lengths, whereas having a ridiculously small mapsize will compromise the very essence of SLAM. One must find the perfect equilibrium between precision on position and reasonable computing speed.

Annexes

```
function [e , E_r, E_p] = observe(r, p)

%   In:
%       r :     robot frame     r = [r_x ; r_y ; r_alpha]
%       p :     point in global frame
%   Out :
%       e :     measurement     e = [range ; bearing]
%       E_r :   Jacobian wrt r
%       E_p :   Jacobian wrt p

if nargout == 1 % No Jacobian requested
    e = scan(toFrame(r,p));

else %  Jacobians requested
    [pr,PR_r,PR_p] = toFrame(r,p);
    [e,E_pr] = scan(pr);

    E_r = E_pr*PR_r;
    E_p = E_pr*PR_p;
end

end
```